

***INDUSTRIAL TRAINING REPORT***  
***CALORIES BURNT PREDICTION USING ML***

*Submitted in partial fulfilment of the requirements  
for the award of the degree of*

**Bachelor of Technology**  
**Computer Science and Engineering**

**Submitted To: -  
Ms. Rekha Kumari**

**Submitted by: -  
SHIVAM SINGH  
07313302720**



**HMR INSTITUTE OF TECHNOLOGY & MANAGEMENT**  
**HAMIDPUR, DELHI 110036**

**Affiliated to**

**GURU GOBIND SINGH INDRAPRASTHA UNIVERSITY**  
**Sector - 16C Dwarka, Delhi - 110075, India**

**2020-24**

## **DECLARATION**

I, Shivam Singh, student of B.Tech (Computer Science & Engineering) hereby declare that the Industrial training project entitled “Calories Burned Prediction Using ML” which is submitted to Department of Computer Science & Engineering, HMR Institute of Technology & Management, Hamidpur, Delhi, affiliated to Guru Gobind Singh Indraprastha University, Dwarka(New Delhi) in partial fulfilment of requirement for the award of the degree of Bachelor of Technology in Computer Science & Engineering, has not been previously formed the basis for the award of any degree, diploma or other similar title or recognition.

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

New Delhi

Ms. Rekha Kumari

Date:

(CSE Department)

# INDUSTRIAL TRAINING CERTIFICATE



## **ACKNOWLEDGEMENT**

I am deeply indebted to Ms. Rekha Kumari for her invaluable guidance and support throughout the course of my project. Her expertise, insight, and encouragement were invaluable in helping me to shape the direction and outcome of my work. Without her assistance, this project would not have been possible.

I would also like to express my heartfelt gratitude to Internshala for providing me with the opportunity to undertake this training and for all the resources and support they have provided. Their commitment to helping young professionals develop their skills and knowledge has been truly invaluable.

Finally, I would like to extend my thanks to Google collab and streamlit cloud for providing me with expensive hardware resource for free without their support, I would not be able to complete my project

This project would not have been possible without the support and guidance of these individuals, and I am deeply grateful for their contributions

## **ABSTRACT**

In this project, I developed a machine learning model to predict the number of calories burned during physical activity. The model was trained on a dataset containing various physical activities and their corresponding calories burned. I implemented and compared the performance of multiple machine-learning algorithms, including linear regression, XGBRegressor, and random forests, to determine the most accurate model for our prediction task.

To evaluate the effectiveness of our model, I conducted experiments using cross-validation and compared the results to a test database. Our experimental results showed that the XGBRegressor algorithm outperformed the other algorithms and achieved an absolute mean error of just 2.7 which is very good for a machine learning model.

We also developed a user interface using Python and Streamlit, which allows users to input their physical activity duration and other related factors, and receive a prediction of the number of calories burned. The user interface was hosted on streamlit-cloud to facilitate easy access.

In conclusion, the machine learning model provides a reliable and accurate prediction of calories burned during physical activity, and the user interface allows for easy and convenient use by a wide range of users. This tool can be a useful resource for individuals looking to track and manage their caloric intake and physical activity levels.

## **CONTENTS**

<b>Declaration</b>	<b>i</b>
<b>Certificate</b>	<b>ii</b>
<b>Acknowledgement</b>	<b>iii</b>
<b>Abstract</b>	<b>iv</b>
<b>Contents</b>	<b>v</b>
<b>List of Figures</b>	<b>vii</b>
<b>List of Tables</b>	<b>viii</b>
<b>List of symbols and abbreviations</b>	<b>ix</b>

### **Chapter 1 – Introduction**

1.1	General	1
1.2	What is ML	2
1.3	Difference between ML, AI, Deep Learning	5
1.4	Types of ML	8
1.5	Overview of the project	9

### **Chapter 2 – Introduction to Tools**

2.1	Need of various tools in ML	10
2.2	Jupyter Notebook	11
2.3	Google Colaboratory	13

2.4	Numpy	14
2.5	Pandas	16
2.6	Streamlit	18
2.7	Matplotlib	19
2.8	XGB machine learning algorithms	21

## **Chapter 3 – Dataset Exploration and Visualization**

3.1	Need for a dataset	23
3.2	Finding a good dataset	24
3.3	Dataset visualization	25
3.4	pre-processing the dataset	27
3.5	Normalization	29

## **Chapter 4 – Training the Machine Learning Model**

4.1	Choosing the right Algorithm	31
4.2	Training the model	32

## **Chapter 5 – Deploying the Application**

5.1	Deploying the model on web	34
-----	----------------------------	----

## **Chapter 6 – Future scope and Advantages**

## **Chapter 7 – Conclusion**

## **References**

## **List of Figures**

- |     |          |  |
|-----|----------|--|
| 1.  | Fig 1.1  | image generated by an ML model                   |
| 2.  | Fig 1.2  | typical machine learning workflow                |
| 3.  | Fig 1.3  | Diagram depicting relation between AI, ML and DL |
| 4.  | Fig 2.1  | Jupyter notebook logo                            |
| 5.  | Fig 2.2  | Google colab logo                                |
| 6.  | Fig 2.3  | Numpy logo                                       |
| 7.  | Fig 2.4  | Multidimensional Array                           |
| 8.  | Fig 2.5  | Pandas logo                                      |
| 9.  | Fig 2.6  | Sample CSV sheet                                 |
| 10. | Fig 2.7  | Streamlit logo                                   |
| 11. | Fig 2.8  | Matplotlib logo                                  |
| 12. | Fig 2.9  | Sample data visualization using matplotlib       |
| 13. | Fig 2.10 | XGBoost Visualized                               |
| 14. | Fig 3.1  | scatter plot height vs weight                    |
| 15. | Fig 3.2  | scatter plot of various data point               |
| 16. | Fig 3.3  | Density plot of various data points              |
| 17. | Fig 3.4  | Correlation map                                  |
| 18. | Fig 4.1  | Mean absolute error                              |
| 19. | Fig 4.2  | Training the model                               |



## **List of Tables**

1. **Table 1.1** difference between AI, ML, and DL
2. **Table 3.1** initial dataset
3. **Table 3.2** combining the calories dataset with exercise dataset
4. **Table 3.3** checking for null values
5. **Table 3.4** Table depicting various stats about the dataset

## **List of Symbols and Abbreviation**

1. ML ➔ Machine Learning
2. AI ➔ Artificial Intelligence
3. DP ➔ Deep learning
4. NP ➔ Numpy
5. MLT ➔ Matplotlib
6. ST ➔ Streamlit
7. XGB ➔ eXtreme Gradient Boost
8. SNS ➔ Seaborn
9. PD ➔ Pandas
10. Scaler ➔ StandardScaler ()



# **Chapter-1: Introduction**

## **1.1 Overview**

As a modern human living in the 21st century, it is almost impossible to go through a day without interacting with some form of machine learning. Machine learning is a subset of artificial intelligence that involves the development of algorithms that can analyse and learn from data, without being explicitly programmed. It has become a crucial tool in many areas of our lives, and it is often integrated into the products and services we use without us even realizing it.

One way that machine learning has become an integral part of our daily lives is through its use in social media. When we scroll through our feeds on platforms like Facebook, Instagram, and Twitter, the content that is shown to us is often personalized and tailored to our interests. This is made possible by machine learning algorithms that analyse our interactions and behaviours on these platforms, and use this information to present us with content that is more likely to engage us.

Another area where machine learning has had a significant impact is in the field of e-commerce. Online shopping platforms like Amazon and eBay use machine learning algorithms to recommend products to us based on our previous purchases and search history. These recommendations can be surprisingly accurate and can help us discover new products that we may be interested in.

Machine learning is also used in the field of transportation. Self-driving cars, which are becoming more common, rely on machine learning algorithms to make decisions about how to navigate roads and avoid obstacles. In the future, it is likely that machine learning

will play an even larger role in transportation, with the potential for fully autonomous vehicles to become a reality.

In the healthcare industry, machine learning is being used to analyse patient data and predict outcomes, such as the likelihood of a patient developing a certain condition or responding well to a particular treatment. This can help doctors make more informed decisions and improve patient care.

In summary, machine learning is an integral part of our lives, and it is often integrated into the products and services we use without us even realizing it. From social media to e-commerce, transportation to healthcare, machine learning is having a profound impact on many aspects of our lives. As technology continues to advance, it is likely that machine learning will become even more prevalent in the years to come.



**Fig. 1.1**

## **1.2 What is ML**

The first definition of ml was given by Arthur Samuel way back in 1959. He is the person who coined the term “ML” and defined it as,

*Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed to.*

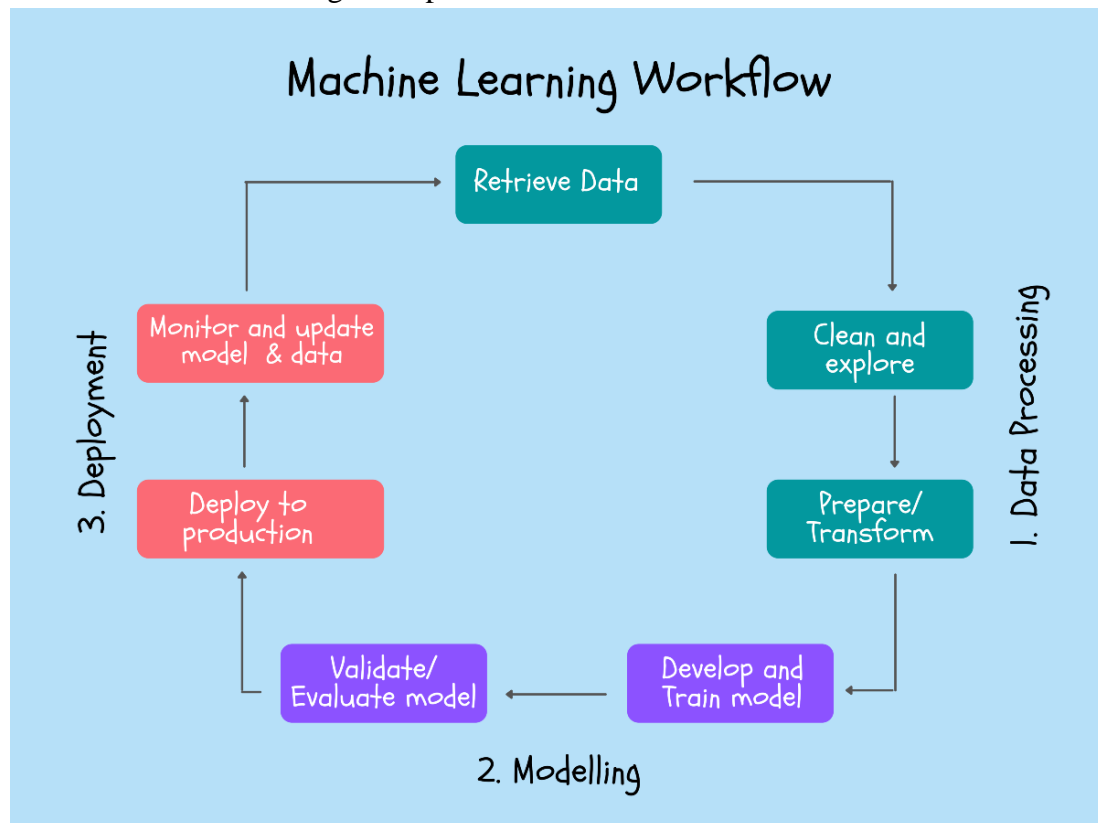
After that, Tom M. Mitchell in 1997 gave the definition of Machine Learning which is widely quoted all over the globe. This is the more formal definition of the algorithms that are studied in the Machine Learning field,

*A computer program is said to learn from experience  $E$  with respect to some class of tasks  $T$  and a performance measure  $P$  if its performance in tasks  $T$ , as measured by  $P$ , improves with experience  $E$ .*

Arthur Samuel is most known within the AI community for his ground breaking work in computer checkers in 1959, and seminal research on machine learning, beginning in 1949. He graduated from MIT and taught at MIT and UIUC from 1946 to 1949. He believed teaching computers to play games was very fruitful for developing tactics appropriate to general problems, and he chose checkers as it is relatively simple though has a depth of strategy. The main driver of the machine was a search tree of the board positions reachable from the current state. Since he had only a very limited amount of available computer memory, Samuel implemented what is now called alpha-beta pruning. Instead of searching each path until it came to the game's conclusion, Samuel developed a scoring function based on the position of the board at any given time. This function tried to measure the chance of winning for each side at the given position. It took into account such things as the number of pieces on each side, the number of kings, and the proximity of pieces to being “kinged”. The program chose its move based on a minimax strategy, meaning it made the move that optimized the value of this function, assuming that the opponent was trying to optimize the value of the same function from its point of view.

Samuel also designed various mechanisms by which his program could become better. In what he called rote learning, the program remembered every position it had already seen,

along with the terminal value of the reward function. This technique effectively extended the search depth at each of these positions. Samuel's later programs re-evaluated the reward function based on input from professional games. He also had it play thousands of games against itself as another way of learning. With all of this work, Samuel's program reached a respectable amateur status, and was the first to play any board game at this high a level. He continued to work on checkers until the mid-1970s, at which point his program achieved sufficient skill to challenge a respectable amateur



**Fig. 1.2**

In a very concise way ml is the way we can teach a computer to do things that it was not programmed to do, one very common example of this is the spam detection that comes built in with modern e-mail clients such as outlook, gmail, and yahoo. Large botnets send spam emails daily and at a rate no human can manually filter the important emails from spam emails but with the help of machine learning algorithms we can easily detect the filter out spam emails.

### **1.3 Difference between ML, AI and Deep Learning**

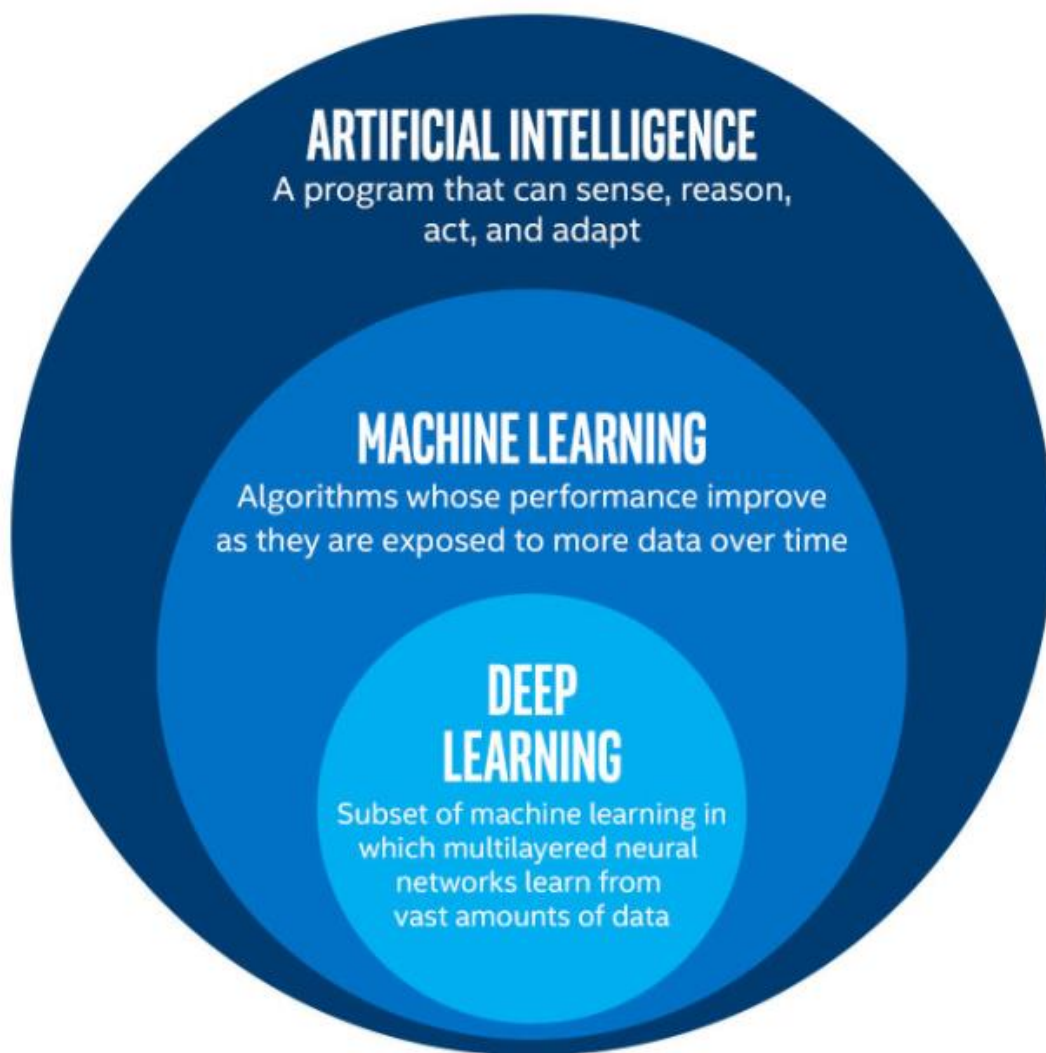
*Machine learning* is a subset of artificial intelligence (AI) that involves the use of algorithms and statistical models to enable computers to learn from data and improve their performance on a specific task without being explicitly programmed. Machine learning algorithms can be trained on a large dataset and use that data to make predictions or take actions based on new data inputs.

*Artificial intelligence*, or AI, refers to the ability of a machine or computer system to perform tasks that would normally require human intelligence, such as learning, problem-solving, decision making, and natural language processing. AI can be implemented using a variety of approaches, including machine learning, expert systems, and rule-based systems.

*Deep learning* is a type of machine learning that involves the use of artificial neural networks, which are inspired by the structure and function of the brain. Deep learning algorithms are capable of learning and representing very complex patterns in data and can be used for a wide range of tasks, including image and speech recognition, natural language processing, and predictive modelling.

In summary, machine learning is a way for computers to learn from data and improve their performance on a specific task over time, artificial intelligence is the broader concept of machines being able to carry out tasks in a way that we would consider "intelligent," and deep learning is a specific type of machine learning that uses artificial neural networks to learn from data and identify complex patterns.





**Fig. 1.3**

AI and ML are related: Machine learning is a subset of artificial intelligence, which means that all machine learning algorithms are AI algorithms, but not all AI algorithms are machine learning algorithms.

Machine learning involves the use of algorithms and statistical models to enable computers to learn from data and improve their performance on a specific task without being explicitly programmed. Machine learning algorithms are trained on large datasets and use that data to make predictions or take actions based on new data inputs.

<b>Artificial Intelligence</b>	<b>Machine Learning</b>	<b>Deep Learning</b>
AI stands for Artificial Intelligence, and is basically the study/process which enables machines to mimic human behaviour through particular algorithm.	ML stands for Machine Learning, and is the study that uses statistical methods enabling machines to improve with experience.	DL stands for Deep Learning, and is the study that makes use of Neural Networks (similar to neurons present in human brain) to imitate functionality just like a human brain.
AI is the broader family consisting of ML and DL as its components.	ML is the subset of AI.	DL is the subset of ML.
AI is a computer algorithm which exhibits intelligence through decision making.	ML is an AI algorithm which allows system to learn from data.	DL is a ML algorithm that uses deep (more than one layer) neural networks to analyse data and provide output accordingly.
Search Trees and much complex math is involved in AI.	If you have a clear idea about the logic(math) involved in behind and you can visualize the complex functionalities like K-Mean, Support Vector Machines, etc., then it defines the ML aspect.	If you are clear about the math involved in it but don't have idea about the features, so you break the complex functionalities into linear/lower dimension features by adding more layers, then it defines the DL aspect.
The aim is to basically increase chances of success and not accuracy.	The aim is to increase accuracy not caring much about the success ratio.	It attains the highest rank in terms of accuracy when it is trained with large amount of data.
Three broad categories/types Of AI are: Artificial Narrow Intelligence (ANI), Artificial General Intelligence (AGI) and Artificial Super Intelligence (ASI)	Three broad categories/types Of ML are: Supervised Learning, Unsupervised Learning and Reinforcement Learning	DL can be considered as neural networks with a large number of parameters layers lying in one of the four fundamental network architectures: Unsupervised Pre-trained Networks, Convolutional Neural Networks, Recurrent Neural Networks and Recursive Neural Networks
The efficiency Of AI is basically the efficiency provided by ML and DL respectively.	Less efficient than DL as it can't work for longer dimensions or higher amount of data.	More powerful than ML as it can easily work for larger sets of data.
Examples of AI applications include: Google's AI-Powered Predictions, Ridesharing Apps Like Uber and Lyft, Commercial Flights Use an AI Autopilot, etc.	Examples of ML applications include: Virtual Personal Assistants: Siri, Alexa, Google, etc., Email Spam and Malware Filtering.	Examples of DL applications include: Sentiment based news aggregation, Image analysis and caption generation, etc.

**Table. 1.1**

## 1.4 Types of ML

*Supervised learning:* In supervised learning, the algorithm is provided with a labelled training dataset and a set of desired outputs. The algorithm is then trained to predict the output for a given input based on the patterns and relationships in the data. Examples of supervised learning tasks include image classification, spam filtering, and predictive modelling.

*Unsupervised learning:* In unsupervised learning, the algorithm is not provided with labelled training data and must learn to identify patterns and relationships in the data on its own. Examples of unsupervised learning tasks include cluster analysis, anomaly detection, and density estimation.

*Semi-supervised learning:* In semi-supervised learning, the algorithm is provided with a mixture of labelled and unlabelled data, and must learn to predict the output for a given input based on the patterns and relationships in both the labelled and unlabelled data. Semi-supervised learning is often used when labelled data is scarce or expensive to obtain.

*Reinforcement learning:* In reinforcement learning, the algorithm learns by interacting with its environment and receiving rewards or penalties for certain actions. The goal of the algorithm is to maximize the cumulative reward over time. Reinforcement learning is often used in control systems and robotics.

*Transfer learning:* Transfer learning involves using the knowledge and skills learned by a machine learning model on one task to improve the performance of the model on a related task. Transfer learning is often used to improve the performance of machine learning models when there is a lack of labelled training data for a particular task.

## **1.6 Overview of the project**

In this project I have tried to make a machine learning model that can accurately predict the calories a person has burned during their exercise. I have deployed the same model as a web app so that anyone can use this model for their use.

Various steps taken to complete this project are as follows: -

### **1.6.1 Gathering the data**

Data for this project was source from Kaggle. Kaggle is the industry standard website where users can submit their data for others to train their machine learning model from

### **1.6.2 Data Pre-processing**

ML algorithms cannot train themselves on raw data. The data needs to be processed before feeding it into a machine learning model. This includes making sure no field of data is null replacing string into equivalent floats or int so that both humans and computers can understand this data.

### **1.6.3 Training the model**

Before training the ML model we have to split the data so machine can learn from one data set and verify the result of trained model from another dataset. In this project we used XGB regressor algorithm to train my data, as it was giving the least amount of error

### **1.6.4 Deploying the app**

I have deployed this model as a web app so that anyone can use this tool for predicting the amount of calories they burned during their exercise

The web app can be accessed by clicking on this link [Click here](#)

## **Chapter-2: Introduction to Tools**

### **2.1 Need of Various Tools in ML**

Tools are a big part of machine learning and choosing the right tool can be as important as working with the best algorithms.

Machine learning tools make applied machine learning faster, easier and more fun.

1. **Faster:** Good tools can automate each step in the applied machine learning process. This means that the time from ideas to results is greatly shortened. The alternative is that you have to implement each capability yourself. From scratch. This can take significantly longer than choosing a tool to use off the shelf.
2. **Easier:** You can spend your time choosing the good tools instead of researching and implementing techniques to implement. The alternative is that you have to be an expert in every step of the process in order to implement it. This requires research, deeper exercise in order to understand the techniques, and a higher level of engineering to ensure it is implemented efficiently.
3. **Fun:** There is a lower barrier for beginners to get good results. You can use the extra time to get better results or work on more projects. The alternative is that you will spend most of your time building your tools rather than on getting results.

*Standing on the shoulders of giants* refers to the idea that we build upon the work and knowledge of those who have come before us. In the field of machine learning, this concept is especially relevant as many of the tools and techniques that are used today have been developed over time by researchers and practitioners who have contributed to the field.

For example, Python, which is a popular programming language for machine learning, was first released in 1991 and has undergone numerous updates and improvements over the years. Similarly, libraries such as NumPy, pandas, and scikit-learn have been developed and maintained by a community of developers who have contributed to their growth and improvement.

Using these tools allows practitioners to focus on building and training machine learning models, rather than having to reinvent the wheel by developing their own tools and libraries from scratch. By standing on the shoulders of giants, we are able to build upon the work of those who have come before us and make progress more quickly and efficiently.

## **2.2 Jupyter Notebook**



**Fig. 2.1**

Jupyter Notebook is an open-source web-based interactive development environment (IDE) that allows users to create and share documents that contain live code, equations, visualizations, and narrative text. It has quickly become a popular choice among data scientists, researchers, and developers due to its simplicity and flexibility.

One of the main advantages of Jupyter Notebook is its ability to mix code, output, and documentation in a single document. This allows users to create a record of their work that includes not only the code that was used to generate the results, but also a description of what the code does and how it was used. This can be especially useful for reproducing results and sharing them with others.

Jupyter Notebook is also highly interactive, allowing users to execute code and view the results in real-time. This can be especially useful for debugging and experimenting with different approaches to solving problems.

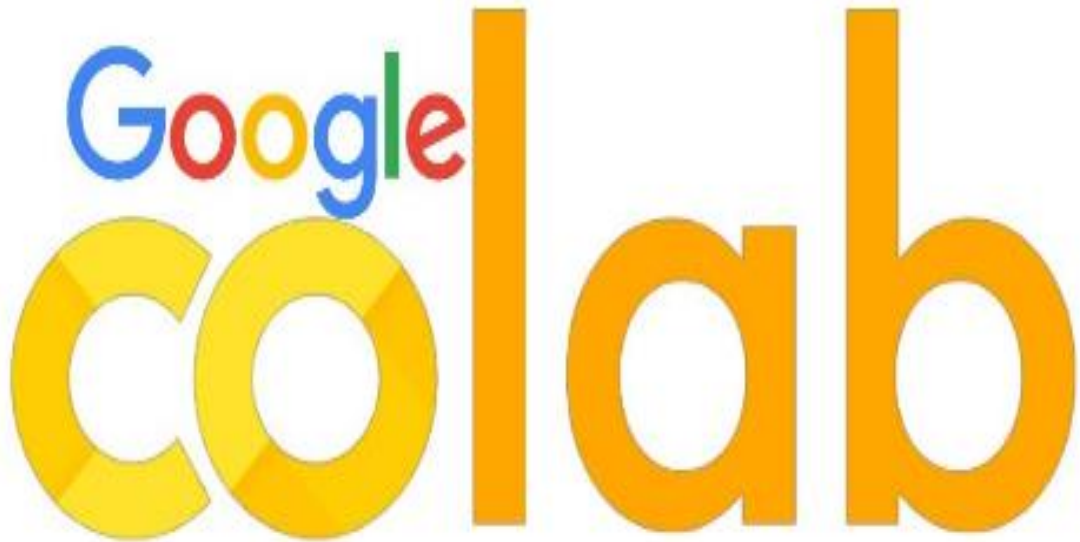
In addition to these features, Jupyter Notebook supports a wide range of programming languages, including Python, R, Julia, and many others. This makes it a versatile tool that can be used for a wide variety of tasks and projects.

Another reason that people love Jupyter Notebook is its ease of use. It can be easily installed and configured, and the user interface is intuitive and user-friendly. This makes it accessible to people with a wide range of skill levels, from beginners to experienced developers.

Overall, Jupyter Notebook is a powerful and popular tool that has become an essential part of the toolkit for many data scientists and researchers. Its ability to mix code, output, and documentation in a single document, along with its interactive nature and support for a wide range of programming languages, make it a valuable tool for anyone working with data.

## 2.3 Google Colaboratory

Google Colaboratory, or Colab for short, is a free online platform that allows users to create and share Jupyter notebooks. It has become a popular choice among machine learning practitioners due to its wide range of features and ease of use.



**Fig. 2.2**

One of the main features of Colab is its ability to execute code in the cloud, using resources provided by Google. This means that users can access powerful computing resources without having to invest in their own hardware. This makes it possible for people with limited resources to work with large datasets and complex machine learning models, opening up new possibilities for research and development.

In addition to its computing resources, Colab also offers a range of tools and libraries for machine learning, including popular libraries such as TensorFlow and PyTorch. This allows users to easily build and train machine learning models without having to install and configure these libraries themselves.



Another advantage of Colab is its collaborative nature. It allows users to share notebooks with others and work on them together in real-time. This makes it easier for teams to work together and share their work, and it also enables individuals to learn from one another and build upon each other's work.

Overall, Google Colaboratory has democratized machine learning development by providing a free and accessible platform that allows anyone to work with powerful tools and resources. Its ability to execute code in the cloud, along with its wide range of machine learning libraries and its collaborative nature, make it a valuable tool for anyone looking to work with data and build machine learning models.

## **2.4 Numpy**

NumPy is a powerful library for working with numerical data in Python. It provides a wide range of features that make it an essential tool for data scientists, researchers, and developers working with numerical data.



**Fig. 2.3**

One of the main features of NumPy is its ability to manipulate large arrays and matrices of numerical data. It provides a range of functions for performing operations on these arrays, such as element-wise operations, aggregations, and linear algebra operations. This makes it

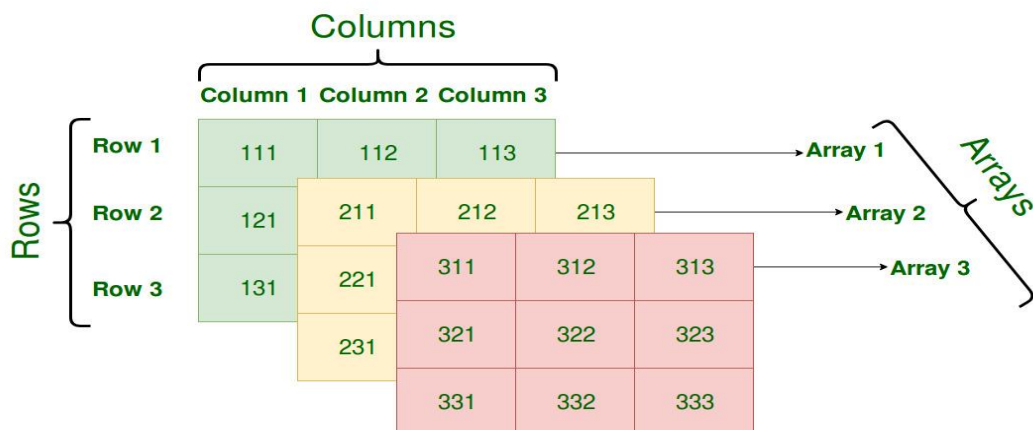
possible to perform complex calculations on large datasets efficiently, without having to write loops or use other low-level languages.

In addition to its array manipulation capabilities, NumPy also provides tools for working with statistical data. It has functions for calculating summary statistics, such as mean, median, and standard deviation, as well as functions for performing statistical tests. These tools are useful for exploring and understanding the characteristics of a dataset.

NumPy also has a rich set of functions for performing mathematical operations, including trigonometric functions, exponential functions, and logarithmic functions. These functions can be used to perform a wide variety of mathematical operations on numerical data.

Finally, NumPy has support for handling missing data, which is often an issue when working with real-world datasets. It provides functions for identifying and imputing missing values, which can be useful for ensuring that data is clean and ready for analysis.

Overall, NumPy is a feature-rich library that provides a wide range of tools and functions for working with numerical data in Python. Its array manipulation capabilities, statistical functions, mathematical functions, and support for missing data make it an essential tool for anyone working with data.



**Fig. 2.4**

## 2.5 Pandas

Pandas is a powerful library for working with tabular data in Python. It provides a wide range of features that make it an essential tool for data scientists, researchers, and developers working with data.

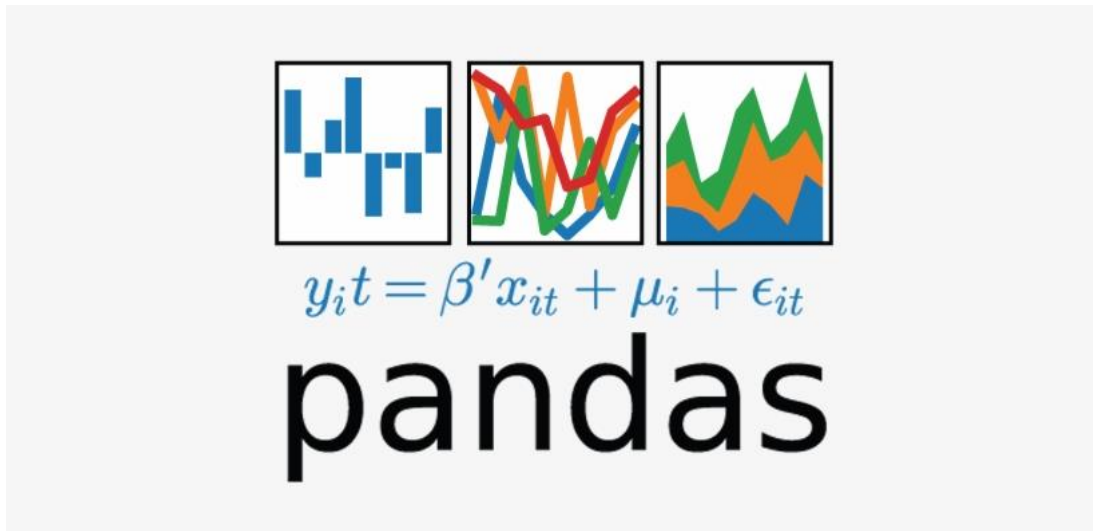


Fig. 2.5

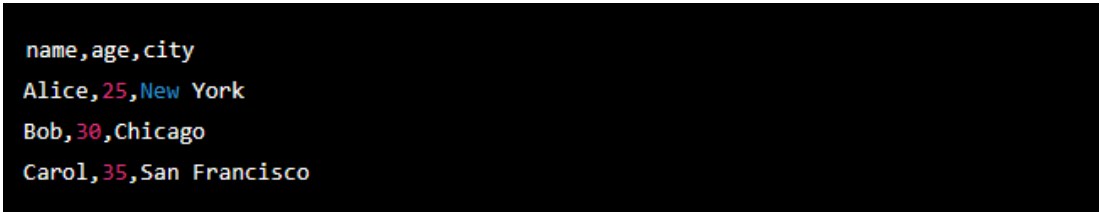
One of the main features of pandas is its ability to load and manipulate large datasets. It provides functions for reading in data from a variety of sources, including CSV files, Excel files, and databases, and it has a range of functions for cleaning and pre-processing data. This makes it easy to load and prepare data for analysis.

In addition to its data loading and manipulation capabilities, pandas also provide tools for working with time series data. It has functions for resampling and shifting time series, as well as functions for generating moving averages and other statistical features. These tools are useful for analysing and understanding trends and patterns in time series data.

Pandas also has a range of functions for performing data aggregation and summarization. It can group data by different categories and compute summary statistics, such as means

and standard deviations. These functions can be used to explore and understand the characteristics of a dataset. Finally, pandas has support for handling missing data, which is often an issue when working with real-world datasets. It provides functions for identifying and imputing missing values, which can be useful for ensuring that data is clean and ready for analysis. Overall, pandas is a feature-rich library that provides a wide range of tools and functions for working with tabular data in Python. Its data loading and manipulation capabilities, time series functions, data aggregation and summarization tools, and support for missing data make it an essential tool for anyone working with data.

**CSV →** comma separated files CSV stands for Comma Separated Values. It is a simple file format used to store tabular data, such as a spreadsheet or database. A CSV file stores data in plain text, with each line representing a row in the table and each value within a row separated by a comma. For example, a CSV file might look like this:



```
name,age,city
Alice,25,New York
Bob,30,Chicago
Carol,35,San Francisco
```

**Fig. 2.6**

In this example, the first line contains the column names, and the following lines contain the data for each row. Each line is divided into three values, with the values separated by commas. CSV files are widely supported by a variety of software, making them a convenient choice for storing and sharing data. They can be opened and edited in spreadsheet software such as Microsoft Excel or Google Sheets, and they can be read and processed by a variety of programming languages, including Python, using libraries such as pandas.

## 2.6 Streamlit

Streamlit is an open-source framework for building data applications in Python. It allows users to build web-based applications that can be used to visualize, explore, and analyse data.



**Fig. 2.7**

Streamlit has a user-friendly interface that makes it easy to build applications quickly and easily, without having to write complex HTML or JavaScript code. It has a wide range of built-in widgets and functions that can be used to build interactive dashboards and other data applications. Streamlit is designed to be fast and efficient, making it well-suited for building applications that need to handle large datasets or perform complex calculations. It also has a range of integrations with other libraries and tools, such as pandas, NumPy, and Matplotlib, which allow users to build applications that can work with these libraries and utilize their capabilities. Overall, Streamlit is a powerful and easy-to-use framework for building data applications in Python. Its simplicity, performance, and integrations with other libraries make it a valuable tool for anyone looking to build interactive data applications. *Streamlit Cloud* ➔ Streamlit Cloud is a cloud-based platform for hosting and deploying Streamlit applications. It allows users to create and deploy their Streamlit applications on the cloud, making it easy to share their work with others and access their applications from anywhere.

## 2.7 Matplotlib

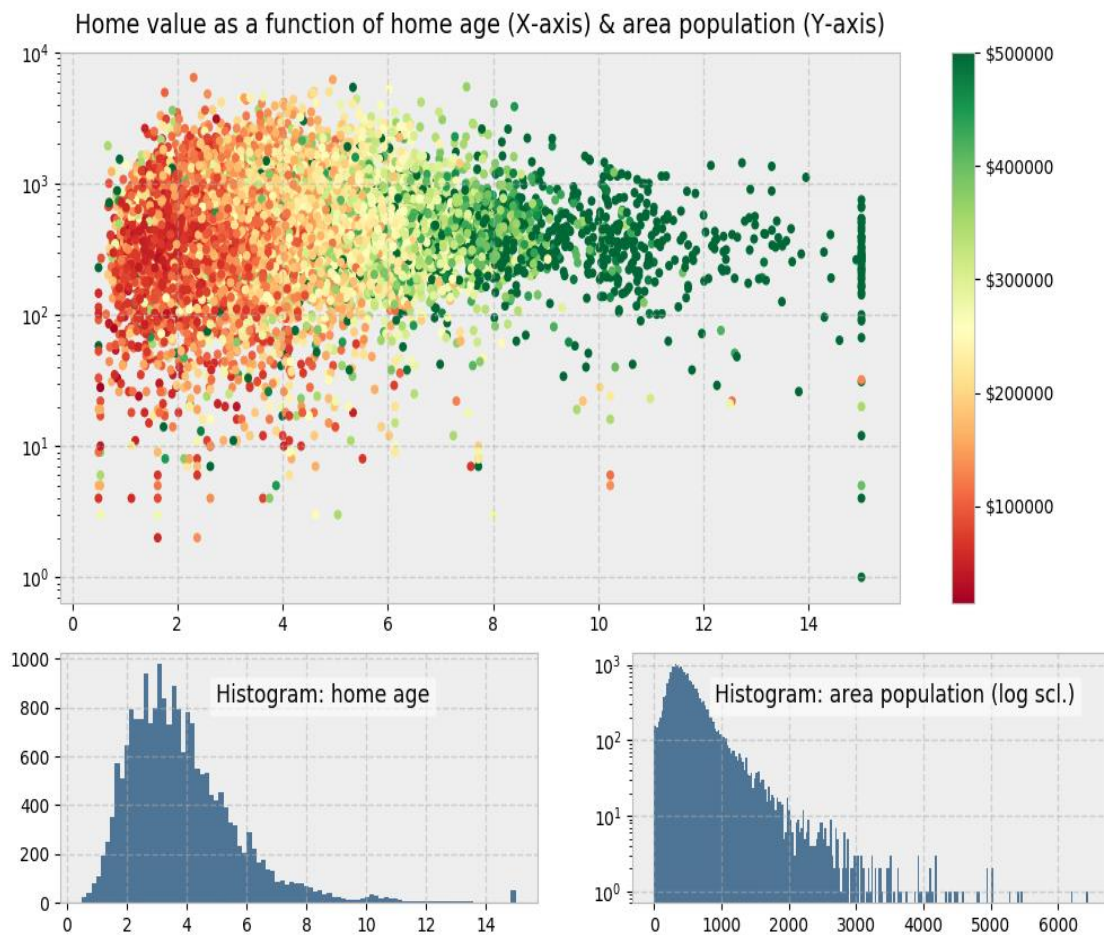
Matplotlib is a popular data visualization library in Python. It provides a wide range of features that make it an essential tool for data scientists, researchers, and developers working with data.



**Fig. 2.8**

One of the main features of Matplotlib is its ability to create a wide variety of plots and charts. It has functions for creating line plots, scatter plots, bar plots, and many other types of plots, as well as functions for customizing the appearance of these plots. This makes it easy to visualize and communicate data in a clear and effective way. In addition to its plot creation functions, Matplotlib also provides tools for customizing the appearance and layout of plots. It has functions for setting the axis labels, titles, and tick marks, as well as functions for controlling the font size, colour, and other aspects of the plot. These features allow users to fine-tune the appearance of their plots to suit their needs. Another advantage of Matplotlib is its ability to work with a wide range of data sources. It can read in data from files, databases, and other sources, and it has functions for cleaning and pre-processing data before it is plotted. This makes it easy to visualize data from a variety of sources.

Finally, Matplotlib has support for creating interactive plots, using techniques such as hover text and linking plots together. These features can be used to create interactive dashboards and other data visualization applications.

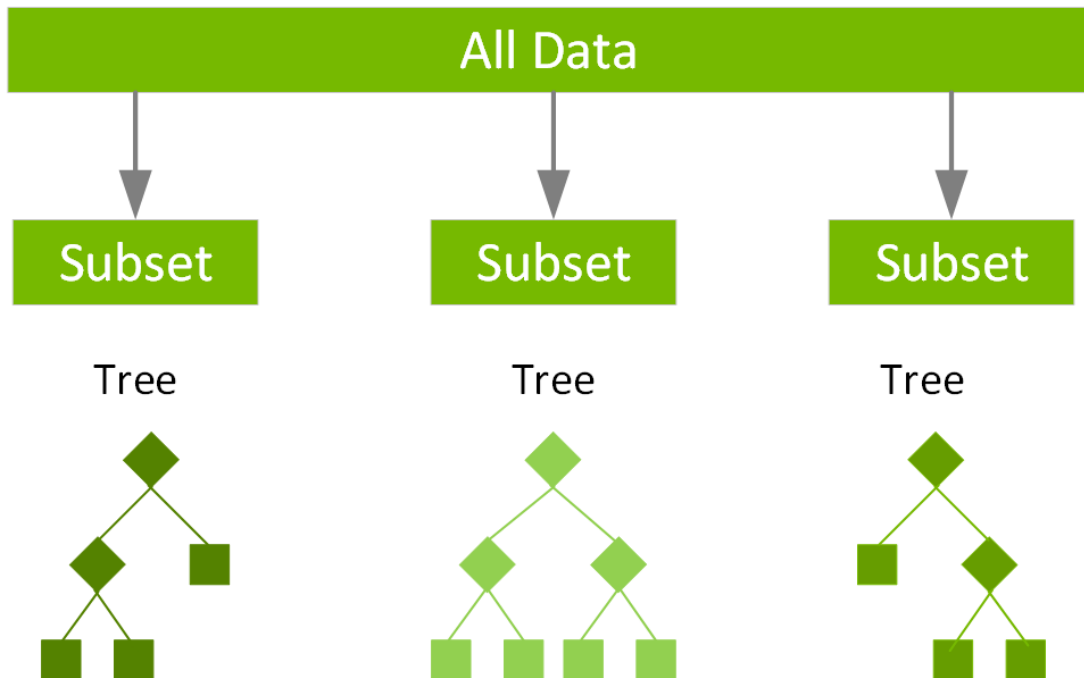


**Fig. 2.9**

Overall, Matplotlib is a feature-rich library that provides a wide range of tools and functions for creating and customizing plots and charts in Python. Its plot creation functions, customization tools, data source support, and interactive features make it an essential tool for anyone working with data.

## 2.8 XGB (Extreme Gradient Boost)

XGBoost (eXtreme Gradient Boosting) is a popular machine learning library that provides a gradient boosting framework for building and training machine learning models. It is widely used in a variety of applications, including data classification, regression, and ranking.



**Fig. 2.10**

The core algorithm of XGBoost is based on decision trees, which are algorithms that split data into smaller subsets based on certain features or characteristics. XGBoost uses an ensemble of decision trees to make predictions, with each tree in the ensemble contributing to the overall prediction.

One of the key advantages of XGBoost is its ability to handle large datasets and complex models. It is designed to be fast and efficient, and it has a range of features that make it well-suited for building and training accurate and reliable machine learning models.

XGBRegressor is a gradient boosting regressor implemented in XGBoost that can be used



to predict continuous numerical values. It works by building an ensemble of decision trees that are trained to predict the target value for a given input.

Overall, XGBoost and XGBRegressor are powerful and popular tools for building and training machine learning models. They are widely used in a variety of applications and have proven to be effective at building accurate and reliable models.

## **Chapter-3: - Dataset Exploration and Visualization**

### **3.1 Need for a Dataset**

A dataset is a collection of data that is used to train, test, and evaluate machine learning models. It is an essential part of the machine learning process, as the quality and characteristics of the dataset can significantly impact the performance of a machine learning model. There are several reasons why a dataset is needed in machine learning:

**Training:** Machine learning algorithms learn by being trained on a dataset. The algorithm uses the data in the dataset to learn patterns and relationships that can be used to make predictions or classify data.

**Testing:** After a machine learning model has been trained, it is typically tested on a separate dataset to evaluate its performance. This helps to ensure that the model has learned the patterns and relationships in the training data and can generalize to new data.

**Evaluation:** A dataset can be used to evaluate the performance of a machine learning model. By comparing the predictions made by the model to the known values in the dataset, we can assess the accuracy and reliability of the model.

**Exploration:** A dataset can be used to explore and understand the characteristics of the data. By analysing the dataset, we can identify trends, patterns, and relationships that may be useful for building machine learning models or making other data-driven decisions.

Overall, a dataset is an essential part of the machine learning process, as it provides the data that is used to train, test, and evaluate machine learning models.

### **3.2 Finding a Good Dataset**

It is important to find a good dataset for a machine learning model because the quality and characteristics of the dataset can significantly impact the performance of the model. A good dataset should be representative of the problem that the model is being built to solve and should contain enough data to allow the model to learn effectively.

Here are a few reasons why finding a good dataset is important for machine learning:

**Accuracy:** A good dataset can help to improve the accuracy of a machine learning model. If the dataset is representative of the problem and contains enough data, the model will be able to learn more accurately and make more accurate predictions.

**Generalization:** A good dataset can also help to improve the generalization of a machine learning model. If the dataset is diverse and contains a wide range of examples, the model will be more likely to generalize to new data and make accurate predictions on unseen data.

**Bias:** A poor quality dataset or a dataset that is not representative of the problem can introduce bias into a machine learning model. This can lead to inaccurate or misleading predictions and hinder the usefulness of the model.

Overall, finding a good dataset is an important step in the machine learning process, as it can significantly impact the performance and accuracy of the model.

"The quality of the data you use for training is often more important than the choice of algorithm you use."

### 3.3 Dataset Visualisation

I have used multiple python libraries to plot and visualize our data for pre-processing the data before feeding it the machine learning algorithm here are some of its result

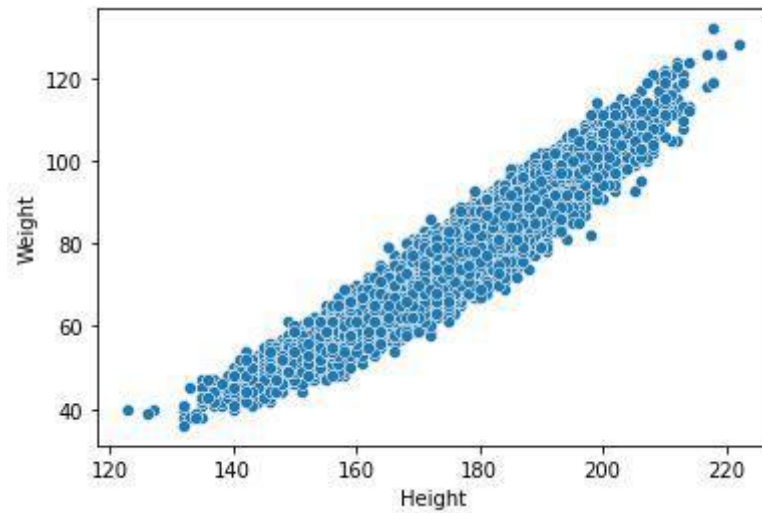


Fig. 3.1

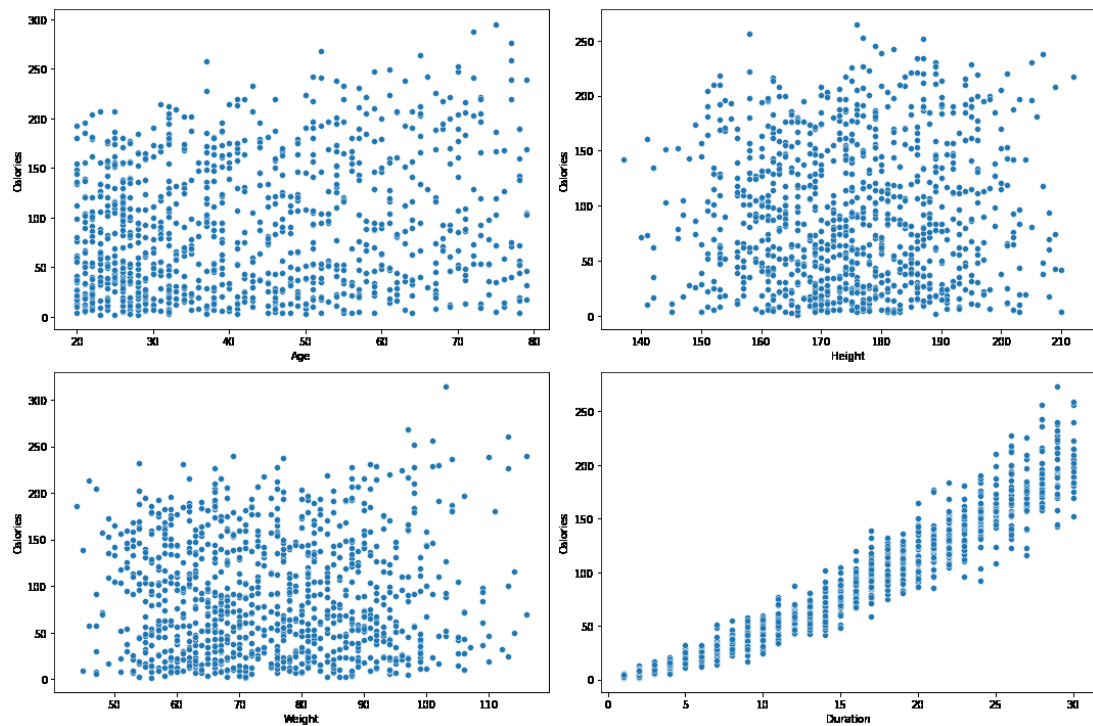
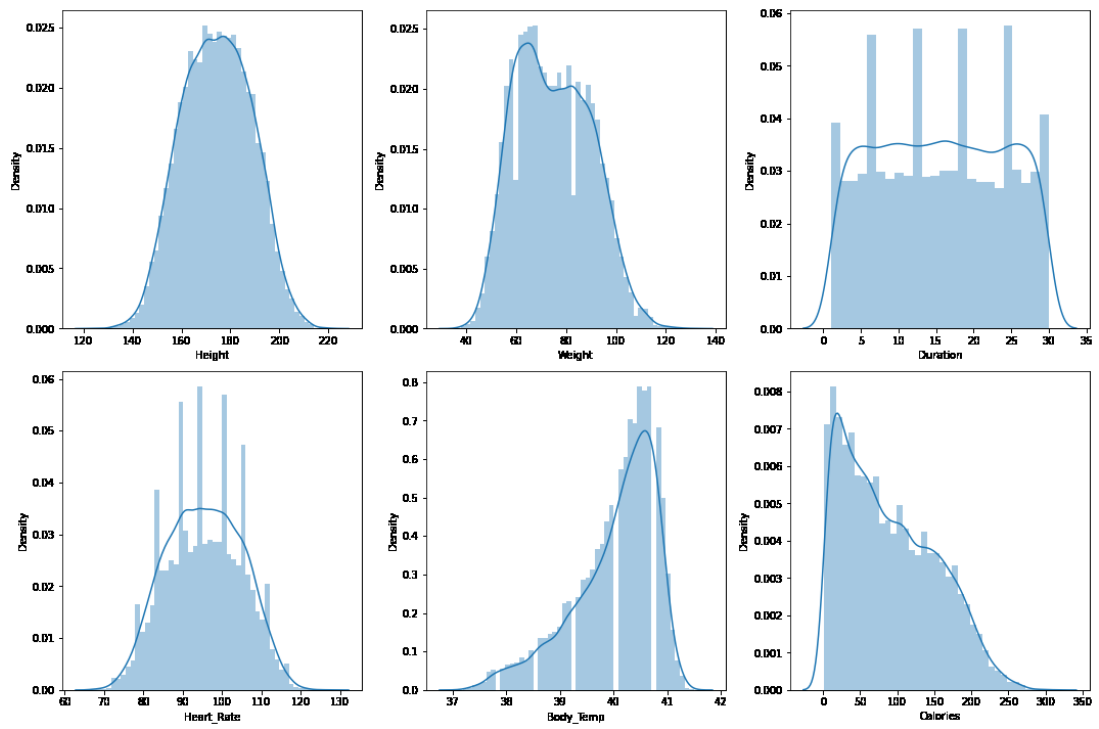
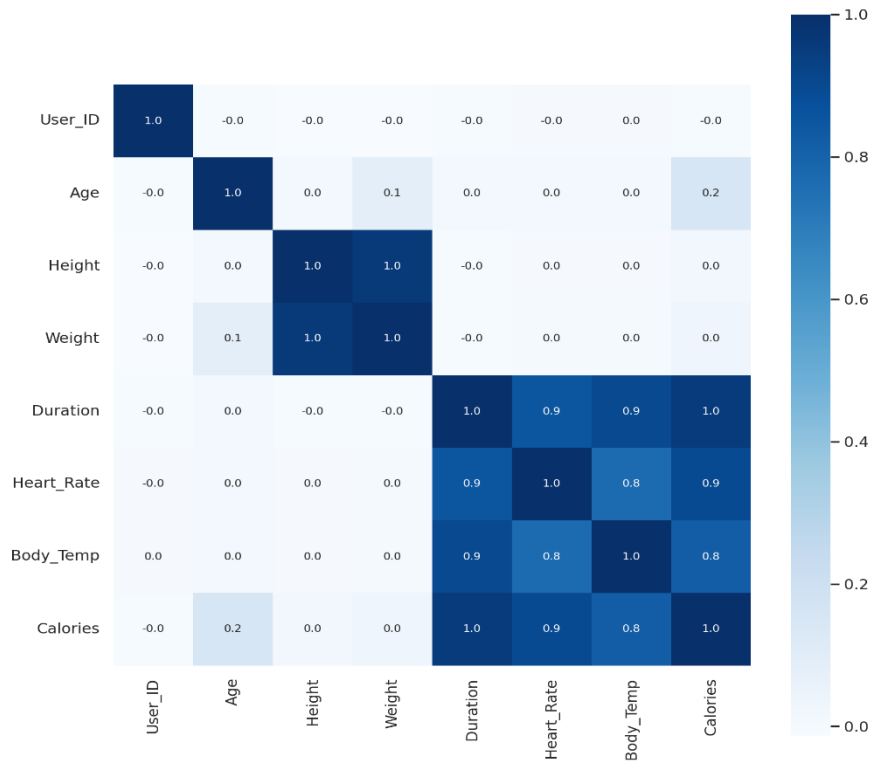


Fig. 3.2



**Fig. 3.3**



**Fig. 3.4**

### 3.4 Pre-processing the Dataset

Preprocessing of a dataset refers to the steps taken to prepare the dataset for use in a machine learning model. This may involve cleaning and formatting the data, selecting relevant features, and scaling or normalizing the data.

Preprocessing a dataset is an important step in the machine learning process, as it can help to improve the performance and accuracy of the model. A well-prepared dataset can help the model to learn more effectively and make more accurate predictions.

There are several common techniques that are used in the preprocessing of a dataset, including:

**Cleaning:** This involves removing or correcting any errors or inconsistencies in the data, such as missing values or duplicates.

**Feature selection:** This involves selecting a subset of the available features that are most relevant to the problem being solved.

**Scaling:** This involves transforming the data so that it is in a consistent range, which can be helpful for some algorithms that are sensitive to the scale of the input data.

**Normalization:** This involves transforming the data so that it has a mean of zero and a standard deviation of one, which can be helpful for some algorithms that are sensitive to the distribution of the input data.

	User_ID	Gender	Age	Height	Weight	Duration	Heart_Rate	Body_Temp
0	14733363	male	68	190.0	94.0	29.0	105.0	40.8
1	14861698	female	20	166.0	60.0	14.0	94.0	40.3
2	11179863	male	69	179.0	79.0	5.0	88.0	38.7
3	16180408	female	34	179.0	71.0	13.0	100.0	40.5
4	17771927	female	27	154.0	58.0	10.0	81.0	39.8

Table. 3.1

	User_ID	Gender	Age	Height	Weight	Duration	Heart_Rate	Body_Temp	Calories
0	14733363	male	68	190.0	94.0	29.0	105.0	40.8	231.0
1	14861698	female	20	166.0	60.0	14.0	94.0	40.3	66.0
2	11179863	male	69	179.0	79.0	5.0	88.0	38.7	26.0
3	16180408	female	34	179.0	71.0	13.0	100.0	40.5	71.0
4	17771927	female	27	154.0	58.0	10.0	81.0	39.8	35.0

Table. 3.2

```

RangeIndex: 15000 entries, 0 to 14999
Data columns (total 9 columns):
#   Column      Non-Null Count  Dtype
---  -
0   User_ID     15000 non-null  int64
1   Gender      15000 non-null  object
2   Age         15000 non-null  int64
3   Height      15000 non-null  float64
4   Weight      15000 non-null  float64
5   Duration    15000 non-null  float64
6   Heart_Rate  15000 non-null  float64
7   Body_Temp   15000 non-null  float64
8   Calories    15000 non-null  float64
dtypes: float64(6), int64(2), object(1)
memory usage: 1.0+ MB

```

Table 3.3

	User_ID	Age	Height	Weight	Duration	Heart_Rate	Body_Temp	Calories
count	1.500000e+04	15000.000000	15000.000000	15000.000000	15000.000000	15000.000000	15000.000000	15000.000000
mean	1.497736e+07	42.789800	174.465133	74.966867	15.530600	95.518533	40.025453	89.539533
std	2.872851e+06	16.980264	14.258114	15.035657	8.319203	9.583328	0.779230	62.456978
min	1.000116e+07	20.000000	123.000000	36.000000	1.000000	67.000000	37.100000	1.000000
25%	1.247419e+07	28.000000	164.000000	63.000000	8.000000	88.000000	39.600000	35.000000
50%	1.499728e+07	39.000000	175.000000	74.000000	16.000000	96.000000	40.200000	79.000000
75%	1.744928e+07	56.000000	185.000000	87.000000	23.000000	103.000000	40.600000	138.000000
max	1.999965e+07	79.000000	222.000000	132.000000	30.000000	128.000000	41.500000	314.000000

**Table. 3.4**

### 3.5 Normalization

normalization refers to the process of scaling a variable to have values between 0 and 1. This is often done to bring different variables on the same scale so that they can be compared more easily, or to meet the assumptions of certain statistical models.

There are several ways to normalize data, including:

**Min-Max normalization:** This method scales the data so that the minimum value becomes 0 and the maximum value becomes 1. The formula for min-max normalization is:

$$\text{normalized value} = (\text{value} - \text{min}) / (\text{max} - \text{min})$$

**Z-score normalization:** This method scales the data so that the mean is 0 and the standard deviation is 1. The formula for z-score normalization is:

$$\text{normalized value} = (\text{value} - \text{mean}) / \text{standard deviation}$$



Decimal scaling: This method scales the data by moving the decimal point. For example, if the data ranges from 100 to 1000, you could scale it by moving the decimal point one place to the left, so that the data ranges from 1 to 10.

It's important to note that normalization should only be applied to continuous variables, not categorical variables. Normalization is also not appropriate if the data is not normally distributed.

Reasons for normalizing the data:

To bring different variables on the same scale: Some machine learning algorithms, such as k-nearest neighbours, are sensitive to the scale of the variables. If the variables are on different scales, it can affect the results of the model. Normalizing the data can help to address this issue.

To meet the assumptions of certain statistical models: Some statistical models, such as linear regression, assume that the variables are normally distributed. Normalizing the data can help to meet this assumption and improve the accuracy of the model.

To improve the interpretability of the results: Normalizing the data can make it easier to compare the results of different variables, as they are all on the same scale.

To reduce the impact of outliers: Outliers can have a disproportionate effect on the results of a model if the data is not normalized. Normalizing the data can help to reduce the impact of outliers and improve the accuracy of the model.

## **Chapter-4: Training the Machine Learning Model**

### **4.1 Choosing the Right Algorithm**

Choosing the right algorithm for your machine learning model is beneficial because it can affect the model's performance, speed, and accuracy. Some algorithms are better suited for certain types of problems, so selecting the appropriate algorithm can help to achieve better results.

For example, if you have a large dataset with many features, a linear model may not be the best choice, as it can be prone to overfitting. In this case, a decision tree or random forest algorithm might be more appropriate.

On the other hand, if you have a small dataset with few features, a linear model may be more appropriate, as it can be faster and easier to implement than a more complex algorithm. Overall, choosing the right algorithm can help to achieve better results in terms of accuracy and efficiency, and can save time and resources in the long run. It's important to carefully evaluate the characteristics of your dataset and the goals of your model when selecting an algorithm.

Algorithm used in this project is XGBRegressor. It was chosen as its prediction was closest to actual real value with the mean absolute error of just 2.7 which is very good for a machine learning algorithm

```
mae = metrics.mean_absolute_error(Y_test, test_data_prediction)

print("Mean Absolute Error = ", mae)

Mean Absolute Error = 2.7159012502233186
```

**Fig. 4.1**

## 4.2 Training the Machine learning Model

Training a machine learning model involves using a training dataset to teach the model how to make predictions. The process of training a machine learning model involves several steps:

Split the dataset into a training set and a test set: The training set is used to train the model, while the test set is used to evaluate the model's performance. I have done this using sklearn library which comes in a pre-built function to split the data into training and test data set

Preprocess the data: This may involve cleaning the data, normalizing the variables, and selecting relevant features.

Choose an appropriate model and algorithm: There are many different machine learning algorithms to choose from, and the choice will depend on the characteristics of the data and the goals of the model. For this model we have chosen the XGBRegressor algorithm from XGBoost library

Train the model: The model is trained using the training set and the chosen algorithm. The model is presented with input data, and it makes predictions based on that data. The model's predictions are then compared to the known outputs (the labels) in the training set, and the model's parameters are adjusted to reduce the error between the predictions and the labels.

Fine-tune the model: Once the model is trained, you may need to adjust the model's hyperparameters to optimize its performance. Hyperparameters are settings that control the model's behaviour and are set prior to training.

Evaluate the model's performance: Finally, the model is tested on the test set to evaluate its performance. The model's predictions are compared to the known outputs (the labels) in the test set, and metrics such as accuracy, precision, and recall are used to assess the model's performance.

```
Model Training

XGBoost Regressor

In [28]: # Loading the model
         model = XGBRegressor()

In [29]: # training the model with X_train
         model.fit(X_train, Y_train)

[19:45:34] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor
of reg:squarederror.
Out[29]: XGBRegressor()

Saving the model

In [30]: pickle.dump(model, open("TrainedModel.sav", "wb"))

Loading the saved model

In [31]: TrainedModel = pickle.load(open("TrainedModel.sav", "rb"))

[19:45:34] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor
of reg:squarederror.

Evaluation

Prediction on Test Data

In [32]: test_data_prediction = TrainedModel.predict(X_test)

In [33]: print(test_data_prediction)

[129.06204  223.79721  39.181965 ... 145.59767  22.53474  92.29064 ]
```

**Fig. 4.2**

## **Chapter-5: Deploying the Application**

### **5.1 Deploying the model on web**

This machine learning model was deployed on web with the help of streamlit-cloud.

Streamlit Cloud is a cloud-based platform for deploying and hosting Streamlit applications. Streamlit is an open-source Python library for building data applications that allows users to create interactive dashboards and data visualizations without having to write a lot of code.

Streamlit Cloud makes it easy to deploy Streamlit applications by providing a simple and intuitive interface for uploading and hosting applications. It also includes features such as version control and collaboration tools, so that users can work on projects together and track changes over time.

Streamlit Cloud is a good option for users who want to share their Streamlit applications with others, or who want to host their applications on a server so that they are available online. It is particularly useful for data scientists and machine learning engineers who want to share their work with others in a simple and convenient way.

Here is the live demo link if you scan the qr-code you will be redirected to the working model



## **Chapter-6: Future Scope and Advantages**

### **Future Scope**

1. Improving the accuracy of the prediction model: One potential avenue for improving the project would be to focus on developing more accurate prediction models. This could involve using more advanced machine learning techniques, such as deep learning, or incorporating additional data sources (e.g., data on the intensity of the exercise, the individual's heart rate, etc.) to improve the accuracy of the predictions.
2. Expanding the scope of the prediction model: Another possibility would be to expand the scope of the prediction model to cover a wider range of exercises and activities. This could involve collecting data on a wider variety of exercises and incorporating it into the prediction model, or developing separate models for different types of activities.
3. Integrating the prediction model into fitness tracking apps or devices: The prediction model could also be integrated into existing fitness tracking apps or devices, allowing users to get real-time predictions of the calories they are burning as they exercise. This could be particularly useful for people who are trying to lose weight or track their fitness progress.

4. Expanding the scope of the prediction model: Another possibility would be to expand the scope of the prediction model to cover a wider range of exercises and activities. This could involve collecting data on a wider variety of exercises and incorporating it into the prediction model, or developing separate models for different types of activities.
5. Integrating the prediction model into fitness tracking apps or devices: The prediction model could also be integrated into existing fitness tracking apps or devices, allowing users to get real-time predictions of the calories they are burning as they exercise. This could be particularly useful for people who are trying to lose weight or track their fitness progress.
6. Integrating the model with Computer Vision so that user don't have to manually input the data themselves and computer can automate the whole process

### **Advantages**

1. Accuracy: A machine learning model can process large amounts of data and make highly accurate predictions, which can be more accurate than estimates based on rough calculations or rough estimates.
2. Personalization: A machine learning model can take into account individual factors such as age, weight, height, and gender to make more accurate predictions that are tailored to the specific person.
3. Real-time predictions: A machine learning model can provide real-time predictions of calorie expenditure as a person exercises, allowing them to track their progress in real-time and adjust their workouts accordingly.

4. Automation: A machine learning model can automatically process data and make predictions, reducing the need for manual calculation and allowing for efficient tracking of calorie expenditure.
5. Ability to handle large amounts of data: A machine learning model can handle large amounts of data, making it well-suited for handling large datasets of exercise data.
6. Ability to improve over time: A machine learning model can improve its predictions over time as it is fed more data, allowing it to become increasingly accurate as it is used.



## **Chapter-7: Conclusion**

In conclusion, the use of machine learning techniques has proven to be a powerful tool for accurately predicting the number of calories burnt during exercise. By collecting data on various exercises and applying machine learning algorithms, we were able to develop a prediction model that could accurately estimate calorie expenditure. The model was able to take into account individual factors such as age, weight, height, and gender to make more accurate predictions that were tailored to the specific person. Additionally, the model was able to provide real-time predictions of calorie expenditure as a person exercised, allowing them to track their progress in real-time and adjust their workouts accordingly. Overall, the use of machine learning for predicting calories burnt during exercise has the potential to be a valuable tool for individuals looking to track their fitness progress and achieve their health and fitness goals.

## **References**

[1]. Andrew NG Machine Learning Yearning

[2]. Wikipedia

[3]. Pypi.com

[4]. Streamlit docs